09/926067
JC12 PCT/PTO 23 AUG 2001

Specification

Optimized bus connection for acceptance of bus transactions

The invention relates to an optimized bus connection for acceptance of bus transactions according to the preamble of claim 1.

The most diverse bus transactions take place in a processor system. Such bus transactions can be classified as those transactions which must be executed in a strictly logical sequence, and can be classified as those transactions which do not have to be executed in a strictly logical sequence.

Bus connections provided with a temporary store operating according to the FIFO principle are known for acceptance of bus transactions. In the temporary store operating according to the FIFO principle, bus transactions are temporarily stored in the sequence of their arrival in the sequence of their arrival and subsequently read out and executed in corresponding sequence, regardless of whether they must be or do not have to be executed in strictly logical sequence.

During bus transactions, it is frequently necessary to wait for results of other bus transactions, for example in order to be able to operate further with updated parameters. Because of the circumstance that the bus transactions are executed in the sequence of arrival, bus transactions which are independent of such transactions must nevertheless wait until the transactions that arrived earlier have been completed. The overall result is slowing and thus loss of performance of the processor system.

The object of the present invention is to provide an optimized bus connection, by which the working speed of a processor

system is accelerated and thus its performance capability is increased.

Starting from a bus connection of the type cited in the introduction, this object is achieved by an optimized bus connection which is provided with the features of claim 1.

Such a bus connection classifies and typifies the arriving bus transactions and allocates them to respective functional lines disposed in parallel. Depending on the class or type of a transaction, the transactions are temporarily stored in the various functional lines in such a way that, on the one hand, they can be treated according to their class or type and, on the other hand, they are sufficiently separated from one another that an adapted sequence can be selected in the sequence of execution. Contributing to this is the fact that some functional lines have a parallel structure. Thus transactions can be moved to the front with the inventive bus connection, so that waiting times until completion of a previously arrived transaction can be eliminated in many cases. The result is acceleration of the mode of operation and thus increased performance capability of a processor system.

Advantageous embodiments of the invention are subject matter of dependent claims.

Accordingly, not only are transactions pending for execution separated according to whether or not they must be executed according to a strictly logical sequence, but also the transactions that do not have to be executed strictly according to a logical sequence are further separated according to whether they are transactions of the read or write type. Read transactions in particular are determining for the performance of an overall system. They must therefore

be given special priority in execution.

As explained in the foregoing, the inventive bus connection eliminates congestion effects which can occur among transactions pending for execution. In the absence of congestion effects, it is possible that entire functional lines will remain almost empty, because arriving transactions can be executed immediately. In order to save further on time needed to transport transactions through the so-called empty functional lines, advantageous embodiments of the invention are provided with shortcuts which bypass the so-called empty functional lines.

A practical example of the invention will be explained in more detail hereinafter with reference to a drawing. Therein the single figure shows an optimized bus connection according to the invention.

The figure shows a processor bus PB, which is part of a higher-level processor system, not indicated in more detail in the figure.

To processor bus PB there is connected a first temporary store S1, which is known in itself and operates according to the FIFO principle, and which stores the arriving bus transactions in their sequence of arrival.

Following first store S1, there are provided three functional sections I, II and III connected in series, of which first functional section I is responsible for reading out, classifying and typifying, by means of a decoder DK, as fast as possible, the transactions temporarily stored in first store S1. On the basis of the classification, the transactions are classified into those transactions that must be executed

in strictly logical sequence. In addition, the transactions are classified into those transactions that do not have to be executed in strictly logical manner. Within the group of transactions that do not have to be executed in strictly logical sequence, a deeper-level typification is performed as to whether the transactions are transactions of the read type or write type.

Second functional section II receives, according to typification and classification, in one of three further stores S2, S3, S4, each of which is disposed in its own allocated functional line, the transaction processes typified and classified by decoder DK.

The transactions classified in the "execution in strictly logical sequence" class are received in store S2 regardless of whether they correspond to the "write" or "read" type. Since these transaction processes must be executed in strictly logical sequence, a deeper-level subdivision into such types is not useful.

Transactions which correspond to the "write" type have allocated to a first write store SS1, in which the information bits to be written can be received. If an individual processor connected to processor bus PB starts a transaction process of the "execution in strictly logical sequence" class and of the "write" type, it transfers the corresponding transaction process together with the information bits to be written to first functional section I. In this way the transaction process for the individual processor is ended. It can be devoted to other tasks. First functional section I ensures that the transaction process is written into second store S2 and the information bits to be written are written into first write store SS1, each of second functional section II. Second

store S2 is designed according to the FIFO principle, in order to be able thereby to maintain the strictly logical sequence of execution of the transaction processes.

In the present practical example, transaction processes of the "execution not in strictly logical sequence" class and of the "read" type are stored in store S3 of second functional section II. Since the sequence of execution is unrestricted, store S3 is designed according to a parallel structure, from which contents can be extracted optionally. The same is true for store 4 of second functional section II, only in regard to the "write" type with the "execution not in strictly logical sequence" transaction class. Since store S4 is responsible for transaction processes of the "write" type, a second write store SS2 is allocated to this store in a manner corresponding to first write store SS1.

Stores S3 and S4 ensure that, in particular, the transactions of the "execution not in strictly logical sequence" class and of the "read" type can be executed immediately and the transaction processes of the "execution not in strictly logical sequence" class of the "write" type can be executed as soon as possible.

The components of second functional section II allocated to respective functional lines lead to an execution unit AE which is common to the functional lines of second functional section II and which is disposed in third functional section III. With the transactions obtained from the functional lines of second functional section II, execution unit AE provides for serial sequencing for further processing, while taking into consideration the importance of the origin of the transactions from the functional lines. In this way, transactions of the "execution not in strictly logical sequence" class may be

moved ahead of transactions of the "execution in strictly logical sequence" class. The transactions rearranged in more favorable sequence in this way are then forwarded to a system bus SB, which is also part of the higher-level processor system not indicated in more detail in the figure.

Shortcuts KW1 and KW2 permit the transaction processes to jump over individual functional sections. For example, if the successive components of functional sections I and II are empty, a transaction process can travel directly from processor bus PB via shortcut KW1 into execution unit AE of functional section III. If store S2 operating according to the FIFO principle in functional section II is empty, an arriving transaction process can be transferred immediately through the store and directed to execution unit AE of functional section III. In both cases, time is saved for transaction processes which affect through transport.